

Overview

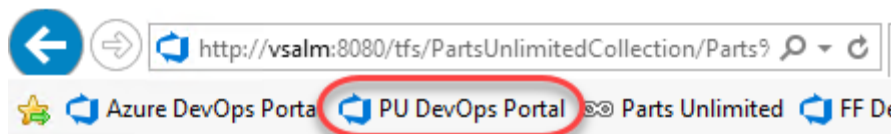
In this lab, you will learn how to use Azure Pipelines to setup a continuous integration (CI) pipeline to build and test your applications. This scriptable build system is both web-based and cross-platform, while also providing a modern interface for visualizing sophisticated workflows. Although we won't demonstrate all of the cross-platform possibilities in this lab, it is important to note that you can also build for iOS, Android, Java (using Ant, Maven, or Gradle), and Linux.

Exercise 1: Build Agent Pools and Queues

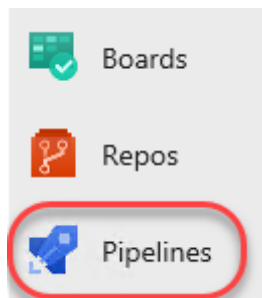
In this exercise, you will learn how to create and configure build agent pools and queues in order to support the new agents in Azure DevOps Server 2019. This new scriptable build system is web-based and cross-platform, and is recommended for all new and existing builds going forward.

Task 1: Getting to build pipelines in Azure DevOps

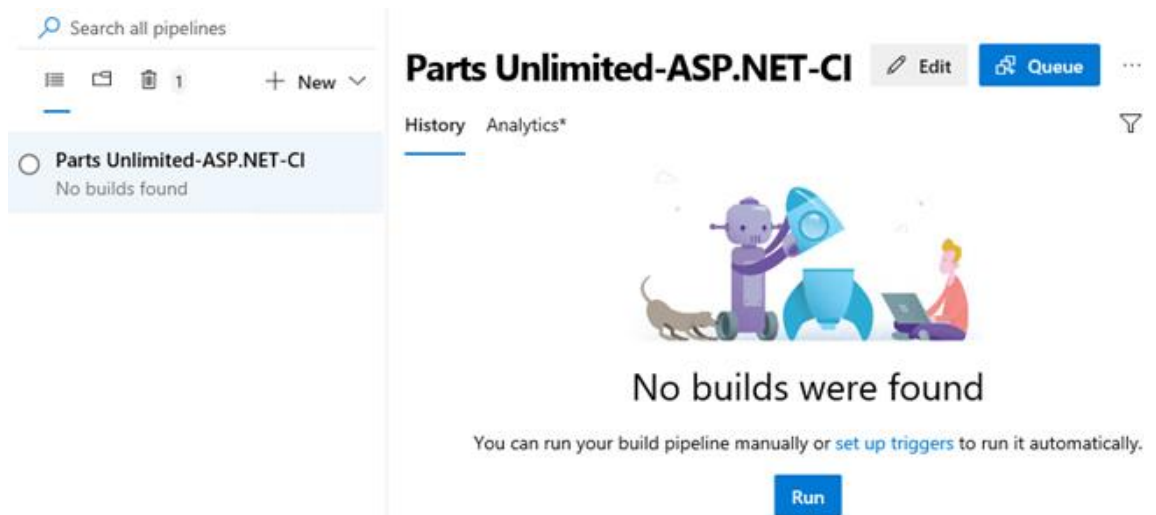
1. Log in
2. Let's get started by touring the **Build** hub in the web portal. Launch **Internet Explorer** from the taskbar and click **PU DevOps Portal** from the favorites bar at the top.



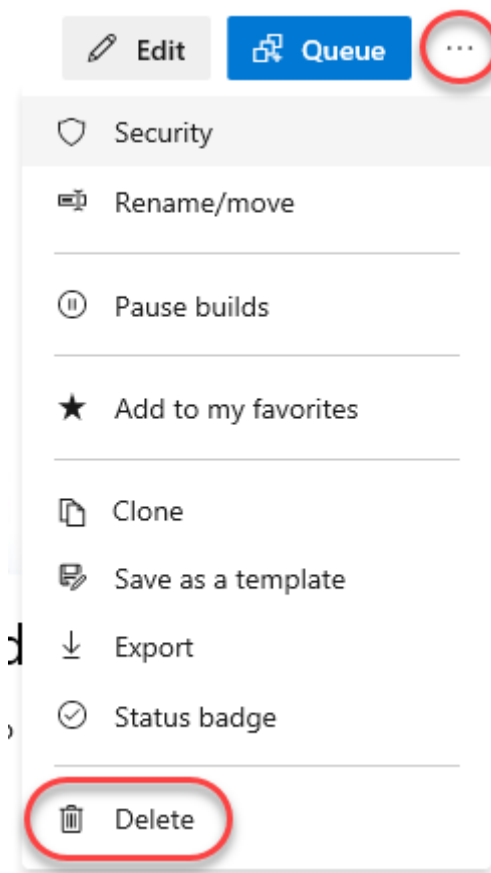
3. Navigate to the **Pipelines** hub.



4. The default view lists build pipelines with the first pipeline selected. The VM ships with a default pipeline, although there are no builds in the history.

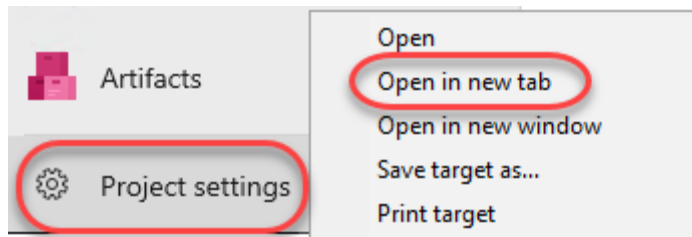


5. From the build pipeline dropdown, select **Delete** and confirm the delete. This lab will focusing on rebuilding this pipeline.

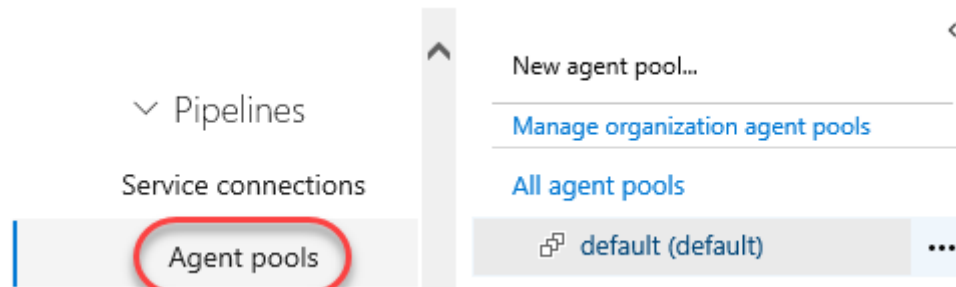


Task 2: Creating an agent pool


1. The first thing that we need to do is to set up an agent pool for the project. This pool can contain both Windows and cross-platform agents. Right-click **Project settings** and select **Open in new tab**.



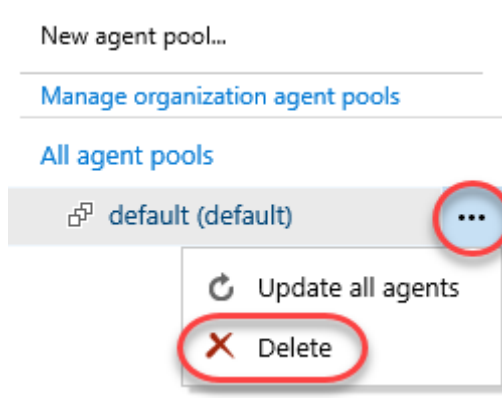
2. Navigate to **Agent pools** under **Pipelines**.



3. There is already an agent pool named **"default"** with a single agent as shown here. For the purposes of this lab, we will delete this pool and add it back in with a new agent. **Delete** the agent.

Agents				
Roles Details Policies				
Enabled	Name	State	Current status	
<input checked="" type="checkbox"/>	VSALM-Build	Online	Idle	

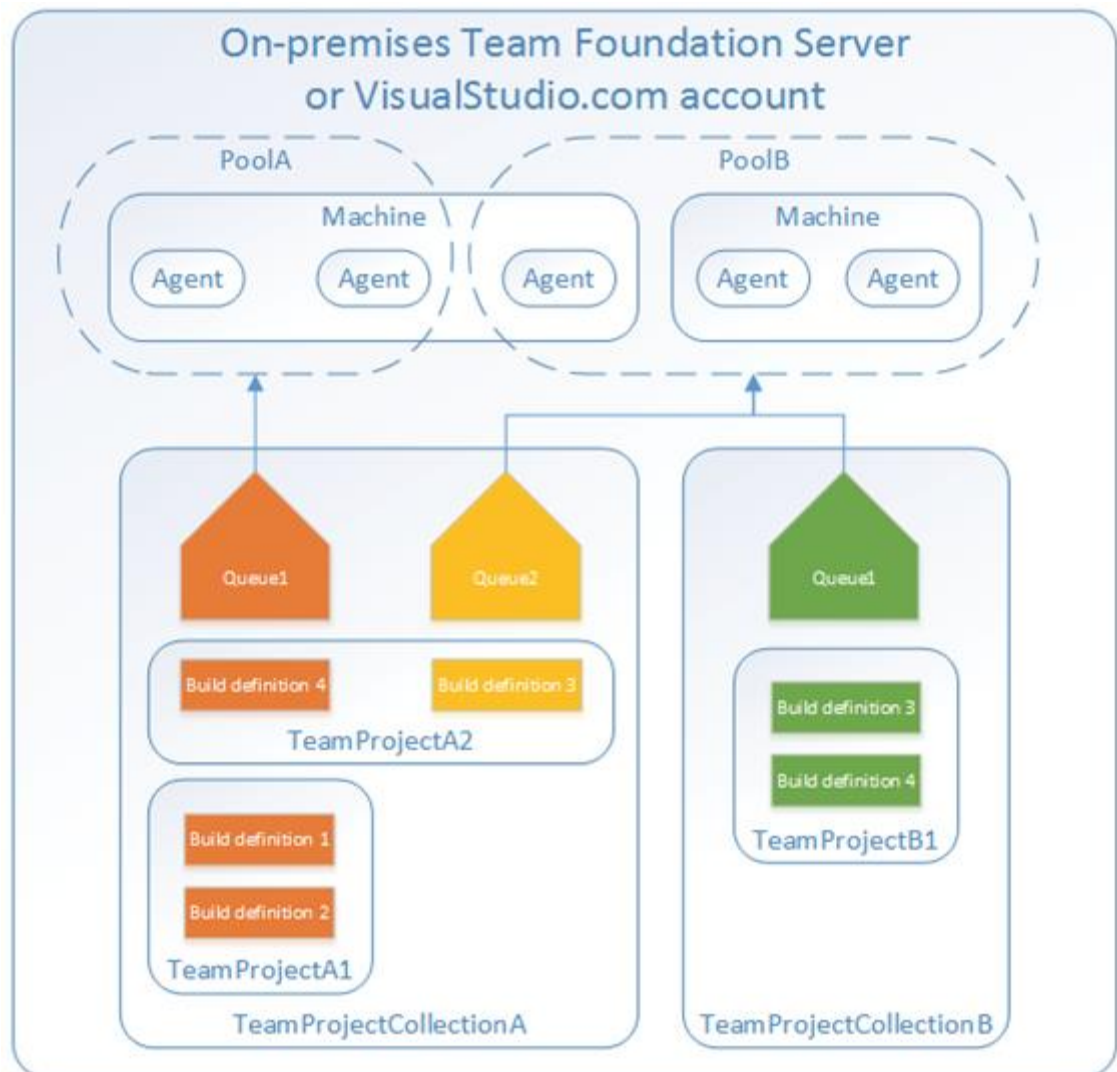
4. **Delete** the pool and confirm.



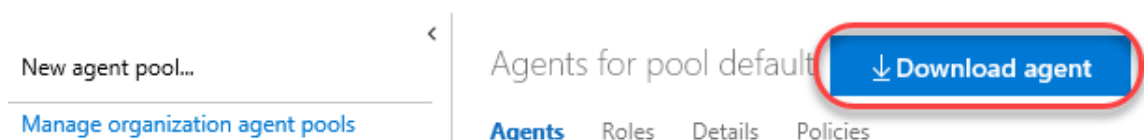
Task 3: Creating a build queue

1. Before we continue with the installation of an agent, let's also ensure that we set up our team project collection with a build **queue** that points to the default agent pool. Since queues are scoped to your team project collection, you can share them across build definitions and team projects.

- This diagram from the MSDN documentation helps to illustrate the relationship between pools, queues, team project collections, and build definitions. Note that you can also install multiple agents on a single machine.



- Click **Download agent**.



- Click **Download** and save the target to disk in a convenient place. This download may take a few minutes, so you can close the **Download agent** dialog and continue to the next step.

Windows

macOS

x86

x64

System prerequisites

Configure your account

Configure your account by following the steps outlined [here](#).

Download the agent

Download

5. Click **New agent pool** to create a new pool.

New agent pool...

Manage organization agent pools

All agent pools

6. Enter a **New pool name** of "**default**" and click **OK**.

Create a project agent pool

- ☐ Base it on an existing organization agent pool
- ☒ Create a new organization and project agent pool

New pool name

default

- ☒ Allow all pipelines to use this pool

OK

Cancel

7. Select the **Release retention** tab. From here, you can specify the default and maximum settings for how long the system retains completed builds. The default retention policy is set at 30 days, with the maximum at 365 days. This means that regardless of what is set on the individual build definition all builds that have not been marked to "Retain indefinitely" will be deleted 30 days after they complete.

Team configuration
GitHub connections
Pipelines
Service connections
Agent pools
Retention
Release retention
Code
Repositories

Retention policy settings

Maximum retention policy

Days to retain a release

Minimum releases to keep

Default retention policy

Days to retain a release

Minimum releases to keep

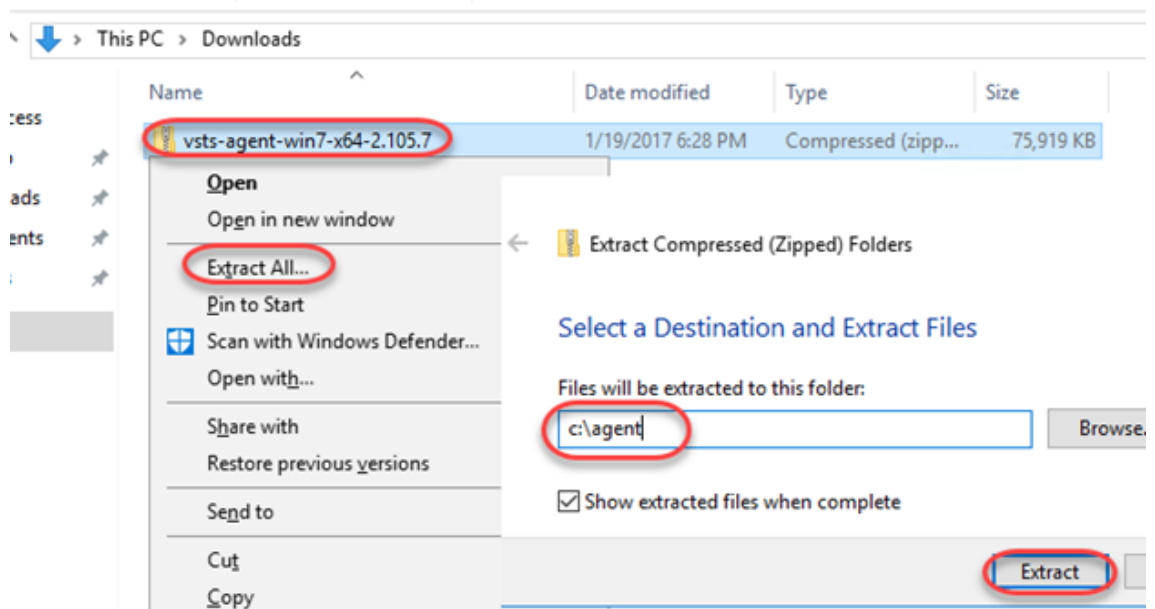
Retain build
☒

Permanently destroy releases

Days to keep releases after deletion

Task 4: Installing and configuring an agent

1. Wait for the agent download to finish if it has not already. Unzip it to **c:\agent** when complete.



2. Launch an instance of **Command Prompt** as **Administrator** from the taskbar.
3. Change to the unzipped agent directory.
4. `cd c:\agent`
5. Execute the agent configuration script.
6. `config.cmd`
7. Enter the server URL "`http://vsalm:8080/tfs`".
8. Press **Enter** for **Integrated** authentication.

9. Press **Enter** to use the default agent pool of "default".
10. Set the agent name to **"VSALM-Build"** and press **Enter**.
11. Press **Enter** to use the default path proposed for the agent work folder "c:\agent_work".
12. When asked if you want to install as a Windows Service, type **"Y"** and then press **Enter**. Note that you could also configure the agent to run in interactive mode, which you may want to do if you were planning to execute coded UI tests.
13. Press **Enter** to run as network service, rather than providing a specific user account.
14. After a few moments, the script should complete with the successful installation and configuration of the new agent.

```
C:\agent>config

>> Connect:

Enter server URL > http://vsalm:8080/tfs
Enter authentication type (press enter for Integrated) >
Connecting to server ...

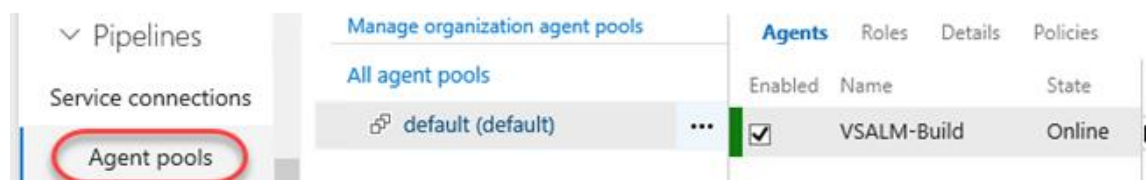
>> Register Agent:

Enter agent pool (press enter for default) >
Enter agent name (press enter for VSALM) > VSALM-Build
Scanning for tool capabilities.
Connecting to the server.
Successfully added the agent
Testing agent connection.
Enter work folder (press enter for _work) >
2019-03-20 19:31:41Z: Settings Saved.
Enter run agent as service? (Y/N) (press enter for N) > y
Enter User account to use for the service (press enter for NT AUTHORITY\NETWORK SERVICE) >
Granting file permissions to 'NT AUTHORITY\NETWORK SERVICE'.
Service vstsagent.vsalm.VSALM-Build successfully installed
Service vstsagent.vsalm.VSALM-Build successfully set recovery option
Service vstsagent.vsalm.VSALM-Build successfully set to delayed auto start
Service vstsagent.vsalm.VSALM-Build successfully configured
Service vstsagent.vsalm.VSALM-Build started successfully

C:\agent>_
```

Note: You weren't prompted for credentials in this case, but under normal circumstances when installing on a remote machine you would be asked to sign in as an agent pool administrator. These credentials are only used once during the configuration process.

15. Return to **Internet Explorer** and navigate to the **Agent pools** tab. The newly created agent will now be in the pool.




16. Select the **Capabilities** tab to take note of the System Capabilities list shown for the agent. System capabilities are name/value pairs that you can use to ensure that your build definition is only run by build agents that meet specified criteria. Environment variables automatically appear in the list. Some additional capabilities (such as .NET Frameworks) are also added automatically. You can also add your own

capabilities to the list based on additional requirements for your builds. Later, when a build is queued, the system sends the job only to agents that have the capabilities demanded by the build definition.

Requests **Capabilities**

USER CAPABILITIES

Shows information about user-defined capabilities

 Add capability

Save changes

Undo changes

SYSTEM CAPABILITIES

Shows information about the capabilities provided by the system

Capability name	Capability value
Agent.ComputerName	VSALM
Agent.HomeDirectory	C:\agent
Agent.Name	VSALM-Build
Agent.OS	Windows_NT
Agent.OSArchitecture	x86

17. Close the browser tab to return to the build pipeline page.

Exercise 2: Working with build pipelines

In this exercise, you will learn how to create a basic build definition from one of the provided templates and then queue the build for execution.

Task 1: Creating a basic build definition from a template

1. Click **New pipeline** to create a new build pipeline.







No build pipelines were found

Automate your build in a few easy steps with a new pipeline.

New pipeline

2. There are a lot of options for selecting the build source, team project, and repo. Accept the defaults and click **Continue**. This will build the master branch of the PartsUnlimited project.

Select a source

 Azure Repos Git	 GitHub Enterprise	 Subversion	 External Git
--	--	---	---

Team project

 Parts Unlimited 

Repository

 PartsUnlimited 

Default branch for manual and scheduled builds

 master 


Continue


3. There are many templates available to build common project types. Everything is customizable, and you can even start with an empty pipeline. Locate the **ASP.NET** template and click **Apply**.

Select a template

Or start with an  **Empty job**

Featured

**ASP.NET**
Build and test an ASP.NET web application.



4. For **Agent pool**, select **default** to use the pool created earlier.

Name *

Parts Unlimited-ASP.NET-CI

Agent pool *




[Pool information](#)

[Manage](#)


default





5. Click the **Add task** button. You can find a wide variety of tasks to cover common scenarios from build through deploy and everything in between. Do not add another task at this time.

Pipeline
Build pipeline

 **Get sources**
PartsUnlimited master

Agent job 1
Run on agent





-  **Use NuGet 4.4.1**
NuGet Tool installer
-  **NuGet restore**
NuGet
-  **Build solution**
Visual Studio Build
-  **Test Assemblies**
Visual Studio Test

Add tasks

Don't see what you need? [Check out the Marketplace.](#)

All Build Utility Test Package Deploy Tool

 **.NET Core**
Build, test, package, or publish a dotnet application, or run a custom dotnet command. For package commands, supports NuGet.org and authenticated feeds like Package Management and MyGet.



by Microsoft Corporation [Learn more](#)

6. Select the **Use NuGet** task. This will ensure the specified version of NuGet is installed.

Pipeline
Build pipeline

Get sources
PartsUnlimited master

Agent job 1
Run on agent

- Use NuGet 4.4.1**
NuGet Tool Installer
- NuGet restore**
NuGet

NuGet Tool Installer

Version: 0.*

Display name: Use NuGet 4.4.1

Version of NuGet.exe to install: 4.4.1

☐ Always download the latest matching version

7. Select **NuGet restore**. This task restores all NuGet packages required by the solution.

Agent job 1
Run on agent

- Use NuGet 4.4.1**
NuGet Tool Installer
- NuGet restore**
NuGet
- Build solution**
Visual Studio Build
- Test Assemblies**
Visual Studio Test
- Publish symbols path**
Index Sources & Publish Symbols

NuGet restore

Display name: NuGet restore

Command: restore

Path to solution, packages.config, or project.json: ***.sln

Feeds and authentication: Feed(s) I select here

8. Select **Build solution**. This will build the solution using the default parameters specified by the ASP.NET template selected earlier.

Agent job 1
Run on agent

- Use NuGet 4.4.1**
NuGet Tool Installer
- NuGet restore**
NuGet
- Build solution**
Visual Studio Build
- Test Assemblies**
Visual Studio Test
- Publish symbols path**
Index Sources & Publish Symbols

Build solution

Display name: Build solution

Solution: ***.sln

Visual Studio Version: Latest

MSBuild Arguments: /p:DeployOnBuild=true /p:WebPublishMethod=Package /p:PackageAsSingleFile=true /p:SkipInvalidConfigurations=true /p:PackageLocation="\$(build.artifactstagingdirectory)\\

9. Select **Test Assemblies**. This task runs project tests based on the configuration. By default, they are detected in the assemblies based on the pattern shown.

Agent job 1
Run on agent

- Use NuGet 4.4.1
NuGet Tool Installer
- NuGet restore
NuGet
- Build solution
Visual Studio Build
- Test Assemblies**
Visual Studio Test
- Publish symbols path
Index Sources & Publish Symbols
- Publish Artifact
Publish Build Artifacts

Display name *

Test Assemblies

Test selection ^

Select tests using * ⓘ

Test assemblies ▾

Test files * ⓘ

`**\$(BuildConfiguration)*test*.dll`
`!**\obj**`

Search folder * ⓘ

`$(System.DefaultWorkingDirectory)`

10. From the **Test platform version**, check for the version of Visual Studio installed. If it is available, select it.

Execution options ^

Select test platform using ⓘ

☒ Version ☐ Specific location

Test platform version ⓘ

Latest ▾

- Latest
- Visual Studio 2017
- Visual Studio 2015
- Installed by Tools Installer

11. If the installed version of Visual Studio is not available, select **Specific location** and enter the path to **vstest.console.exe**. It should be something like "**C:\Program Files (x86)\Microsoft Visual**

Studio\2019\Preview\Common7\IDE\Extensions\TestPlatform\vstest.console.exe".

Execution options ^

Select test platform using ⓘ

☐ Version ☒ Specific location

Path to vstest.console.exe ⓘ

C:\Program Files (x86)\Microsoft Visual Studio\2019
\Preview\Common7
\IDE\Extensions\TestPlatform\vstest.console.exe|

12. Select **Publish symbols path**. This task specifies where and how symbols are pushed.

Use NuGet 4.4.1
NuGet Tool Installer

NuGet restore
NuGet

Build solution
Visual Studio Build

Test Assemblies
Visual Studio Test

Publish symbols path
Index Sources & Publish Symbols

Publish Artifact
Publish Build Artifacts

Index Sources & Publish Symbols ⓘ

Version 2.* ▾

[Link settings](#) [View YAML](#)

Display name *

Publish symbols path

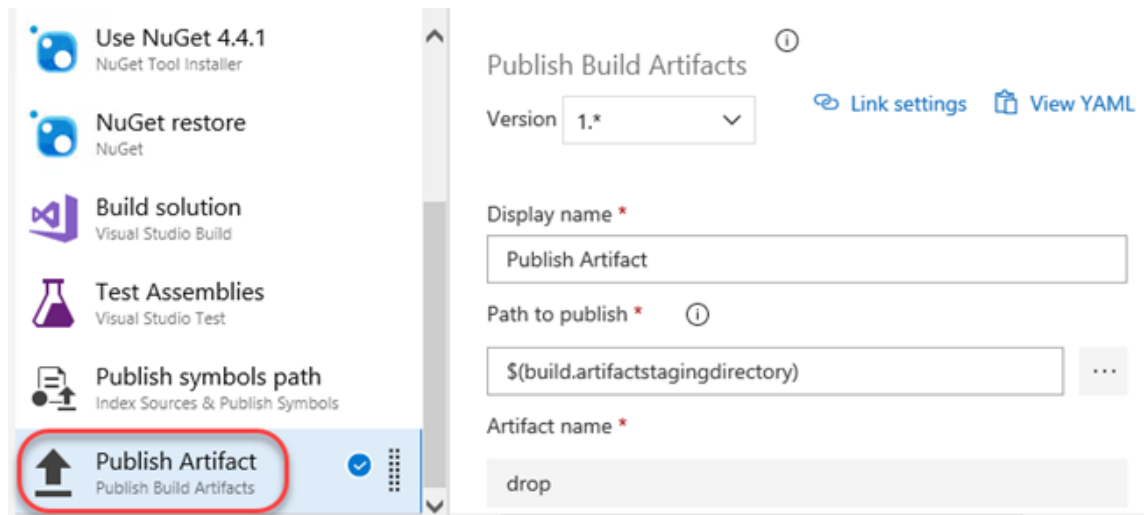
Path to symbols folder ⓘ

\$(Build.SourcesDirectory)

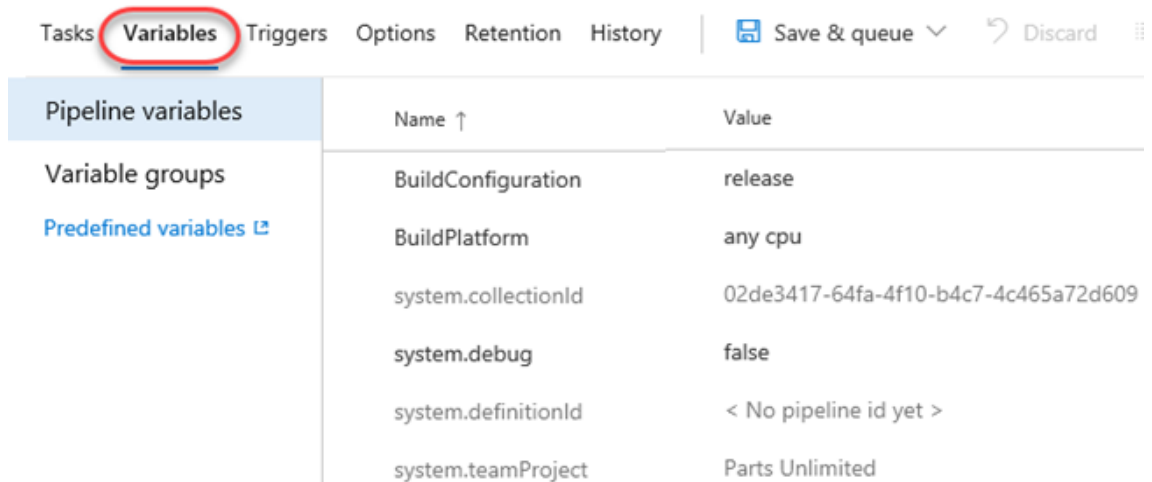
Search pattern * ⓘ

\bin*.pdb

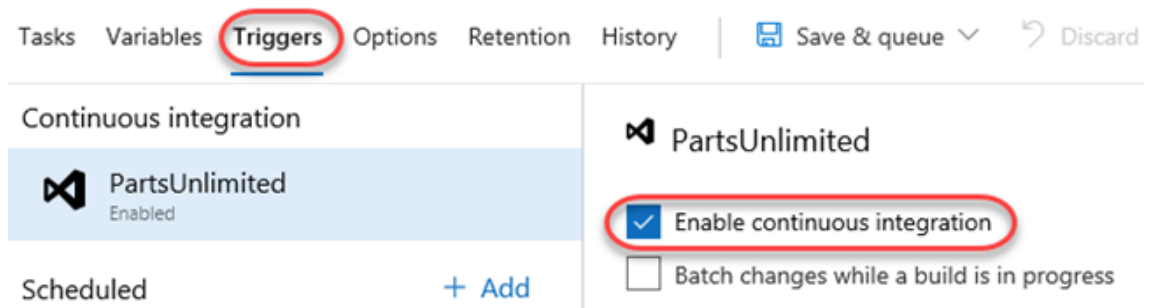
13. Select **Publish Artifact**. This task specifies where and how the project artifact is published.




14. Select the **Variables** tab. This enables you to specify centralized variables to share with the tasks of the pipeline.



15. The **Triggers** tab enables you to define if and when builds are automatically invoked. Check **Enable continuous integration**, which will invoke a build when a change is committed to the master branch. You can also schedule builds, such as if your team runs nightly builds.



16. Select the **Options** tab. This provides a place to set build properties and limits.

Tasks Variables Triggers **Options** Retention History |  Save & queue ▾

Build properties

Define general build pipeline settings

Description

Build number format ⓘ

\$(date:yyyyMMdd)\$(rev:.r)

Build job

Define build job authorization and timeout settings

Build job authorization scope ⓘ




Project collection

Build job timeout in minutes ⓘ

60

Build job cancel timeout in minutes ⓘ

17. The **Retention** tab provides a place to set policies and settings for builds.


Tasks Variables Triggers Options **Retention** History |  Save & queue ▾  Discard 

Policies

[+ Add](#)

Policies are evaluated in order, applying the first matching policy to each build. The default rule at the bottom matches all builds.

Keep for 10 days, 1 good build
+refs/heads/*


 Keep for 30 days, 10 good builds
Maximum

Settings

Branch filters


Type Branch specification

Include ▾

 * ▾


[+ Add](#)

18. Select the **History** tab to view build history.




Tasks Variables Triggers Options Retention **History** | 

Changed By	Change Type	Changed Date	Comment
------------	-------------	--------------	---------

19. From the **Save & queue** dropdown, select **Save**.

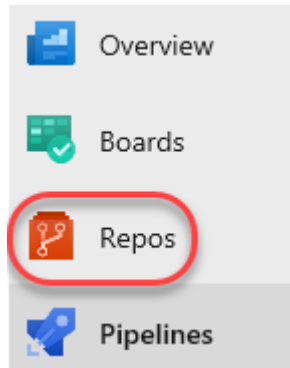
Tasks Variables Triggers Options Retention **History** |  Save & queue ▾

Changed By	Change T
------------	----------

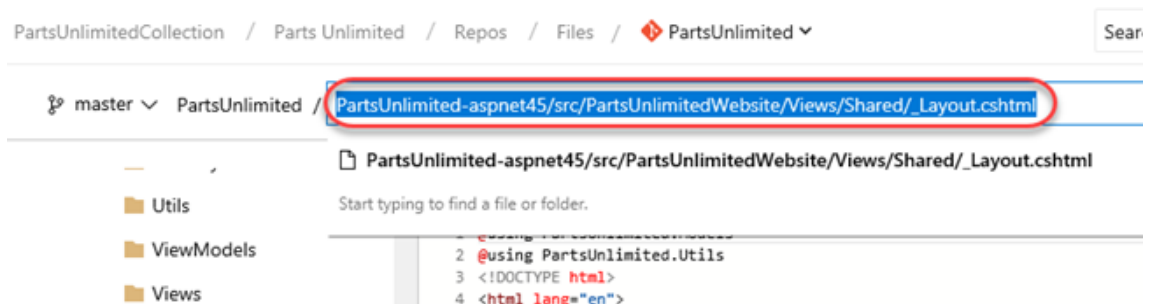
 Save & queue
 **Save**
 Save as draft

Task 2: Triggering and tracking a continuous integration build

1. Since the continuous integration trigger was enabled, you can invoke a build by committing a change to the master branch. Select **Repos**.



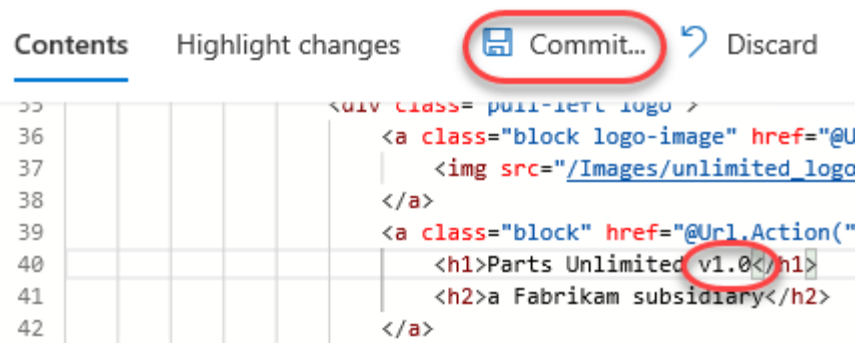
2. Navigate to **PartsUnlimited-aspnet45/src/PartsUnlimitedWebsite/Views/Shared/_Layout.cshtml**.



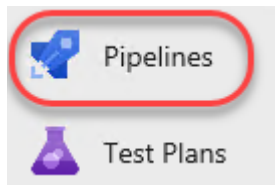
3. Click **Edit**.



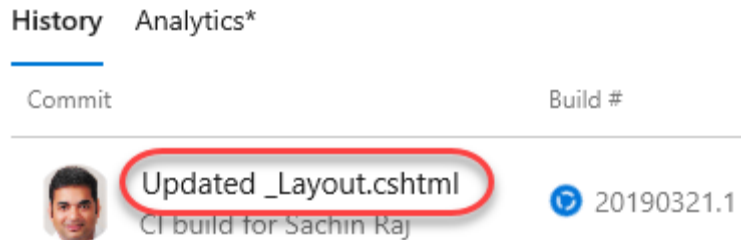
4. Make a cosmetic change by appending "**v1.0**" to the **h1** tag. Click **Commit** and confirm.



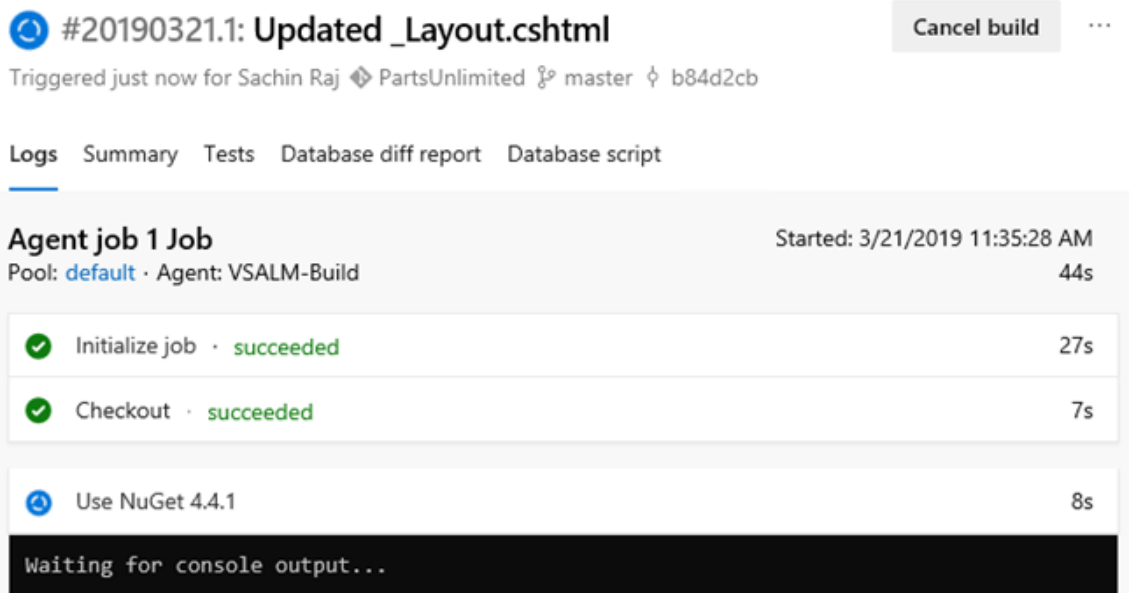
5. Select **Pipelines**.



- There should now be a build running. Select it.



- Follow the build through to completion. Each task is logged for easy tracking.



- The build should succeed. If there are any errors, retrace earlier steps.

✓ #20190321.1: Updated _Layout.cshtml

Triggered today at 11:34 am for Sachin Raj master

[Release](#)[Artifacts](#)

[Logs](#) [Summary](#) [Tests](#) [Database diff report](#) [Database script](#)

Agent job 1 Job

Started: 3/21/2019 11:35:28 AM

Pool: [default](#) · Agent: VSALM-Build

... 2m 23s

✓ Prepare job · succeeded	<1s
✓ Initialize job · succeeded	27s
✓ Checkout · succeeded	7s
✓ Use NuGet 4.4.1 · succeeded	4s
✓ NuGet restore · succeeded	22s
✓ Build solution · succeeded 1 warning	39s
✓ Test Assemblies · succeeded	19s
✓ Publish symbols path · succeeded 1 warning	8s
✓ Publish Artifact · succeeded	11s
✓ Post-job: Checkout · succeeded	<1s
✓ Report build status · succeeded	1s

9. Select the **Summary** tab to review the progression of the lab.

✓ #20190321.1: Updated _Layout.cshtml

Triggered today at 11:34 am for Sachin Raj master

[Release](#)[Artifacts](#)

[Logs](#) [Summary](#) [Tests](#) [Database diff report](#) [Database script](#)

Progression



Deployments

0

No deployments were found for this build.



Build artifacts published

1



drop

File container

Queued

7m ago

Started after 24s in queue

7m ago

Completed after 2m 37s

4m ago

Builds since today at 11:34 am



10. The **Tests** tab provides a summary of the tests run during the build.

✓ #20190321.1: Updated _Layout.cshtml
Triggered today at 11:34 am for Sachin Raj master b84d2cb

Logs Summary **Tests** Database diff report Database script

Summary ^

1 Run(s) Completed (1 Passed, 0 Failed, 0 Not impacted, 0 Others)

16

Total tests

+16



14 Passed

0 Failed

2 Others

87.5%

Pass percentage

↑ 87.5%

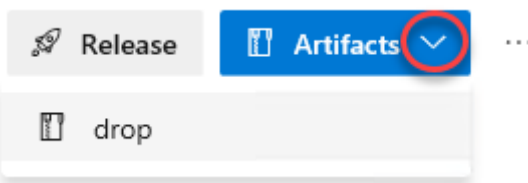
5s 216ms

Run duration ⓘ

↑ +5s 216ms

Bug Link

11. You can also review the build drop itself from the **Artifacts** dropdown.



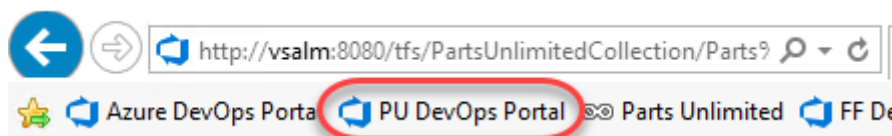
Exercise 2: Continuous Release Management

In this exercise, you will use the release management features of Azure DevOps Server to produce an automated deployment solution. This exercise will take an existing enterprise application and automate its deployment to the development team's testing environment after each source check-in.

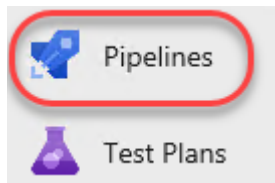
Task 1: Invoking a build

This task will create a build as a starting point for a continuous release. If you already completed the build lab, you can skip to the next task.

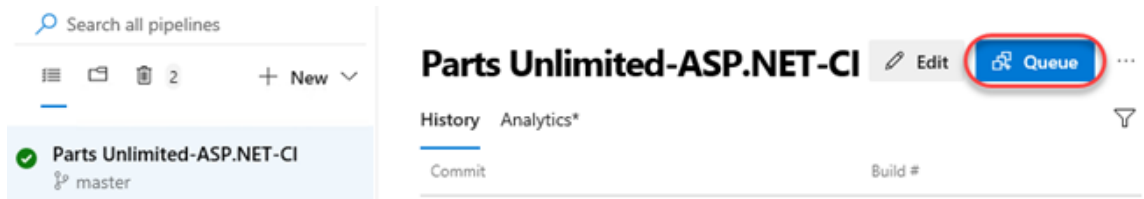
1. Log in as **Sachin Raj (VSALM\Sachin)**. All user passwords are **P2ssw0rd**.
2. Launch **Internet Explorer** from the taskbar and click **PU DevOps Portal** from the favorites bar at the top.



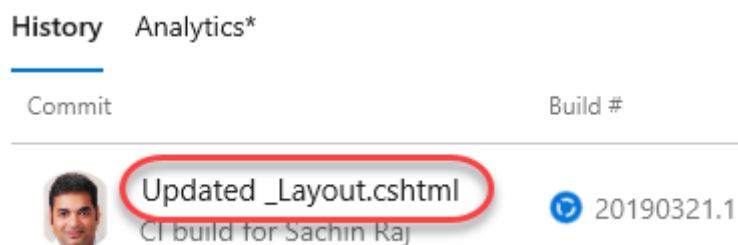
3. Select **Pipelines**.



4. Click **Queue** to invoke a build manually.



5. Select the build to follow its progress through to completion.




Task 2: Creating a continuous release pipeline

1. Once the build has completed, click **Release** to create a release pipeline. Note that you can also create a release pipeline from scratch, but this option will preconfigure the release pipeline to use this build pipeline's output.



2. There are many release pipeline templates available out of the box for common deployment scenarios. To start off with, we will create a stage in the release pipeline that deploys the application to the IIS instance running on the VM. Select the **IIS website deployment** template and click **Apply**. This will provide the tasks required to deploy to IIS.

Select a template

Or start with an  **Empty job**



IIS website and SQL database partially online upgrade

Deployment Group: Upgrade ASP.NET or ASP.NET Core based websites. Upgrade a SQL database using SQL scripts executed when the web application is online followed by SQL scripts executed when the web application is offline. Applicable for physical or virtual machines (VM) deployments.



IIS website deployment

Deployment Group: Deploy an ASP.NET or ASP.NET Core web application to an IIS website on physical or virtual machines (VM).

Apply



Run automated tests from Test Manager

Trigger automated test cases from test plans and suites in Test hub.

- Set the **Stage name** to “**Local IIS**”.

Stage

 Delete  Move  ...

Local IIS

Properties ^

Name and owners of the stage

Stage name

Local IIS



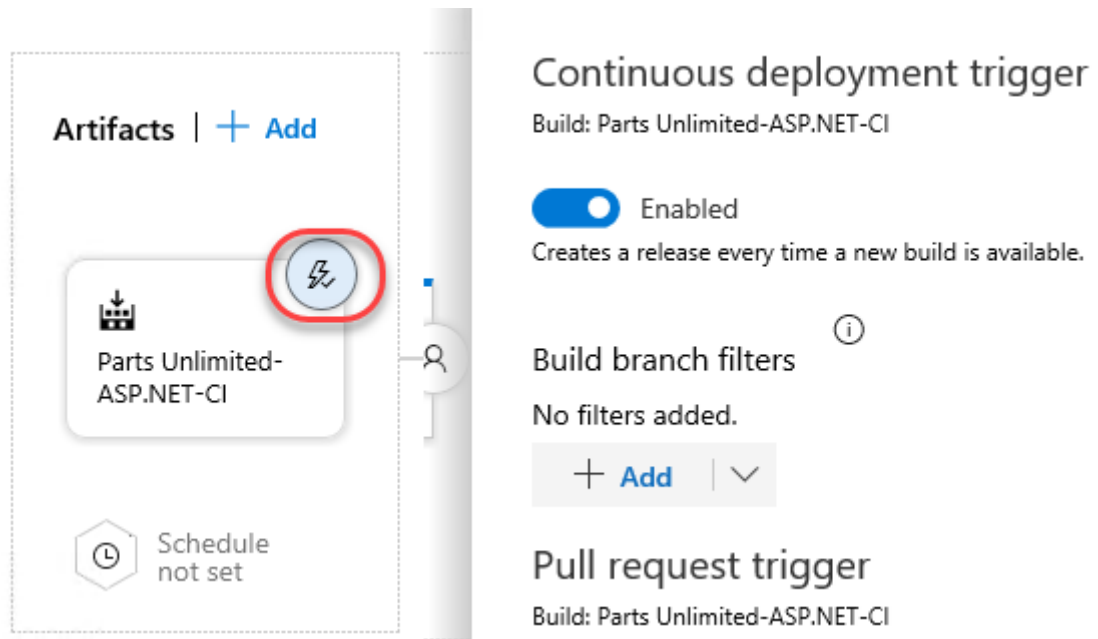
Stage owner



Sachin Raj



- Since this release was created based on a successful build, the artifact details have been preconfigured. Click the **continuous deployment trigger** button on the **Artifacts** box to see that continuous deployment has been enabled so that every new build will invoke this release pipeline. And since the build pipeline is triggered by a master branch commit, any change from a developer can result in the site being updated with minimal overhead.



Artifacts | + Add

Parts Unlimited-ASP.NET-CI

Schedule not set

Continuous deployment trigger

Build: Parts Unlimited-ASP.NET-CI

☒ Enabled
Creates a release every time a new build is available.

Build branch filters

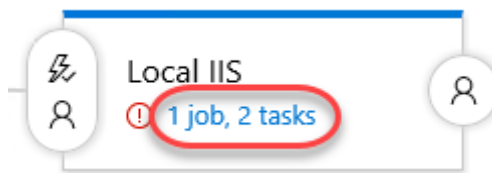
No filters added.

+ Add | v

Pull request trigger

Build: Parts Unlimited-ASP.NET-CI

5. Select **1 job, 2 tasks** under the **Local IIS** stage.



Local IIS

1 job, 2 tasks

6. The site we want to deploy to is hosted on the local IIS machine, so update the **Website name** to "**PartsUnlimited**". Also clear the **Add binding** box since the bindings have already been configured.

Action *

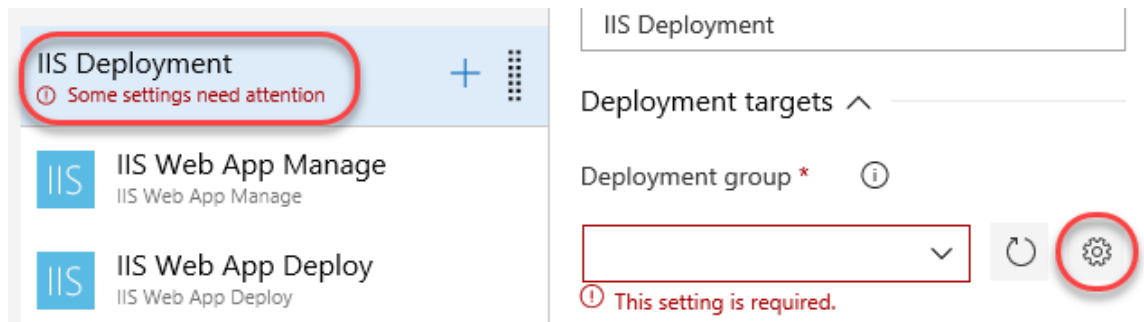
Create Or Update

Website name *

PartsUnlimited

☐ Add binding

7. Select the **IIS Deployment** job definition. In order to deploy, we will need to define a **deployment group**. Click the **Settings** button to open this configuration in a new browser tab.



8. Click **Add a deployment group**.



Add a deployment group

Define a logical group of target machines for parallel deployment.

Add a deployment group

[Learn more about deployment groups](#)

9. Set the **Deployment group name** to “**Local IIS**” and click **Create**.

Deployment group name

Local IIS

Description

Create

10. The provided PowerShell script will do everything you need to download, install, and configure the local machine as a deployment agent for this group. Click **Copy script to the clipboard**.

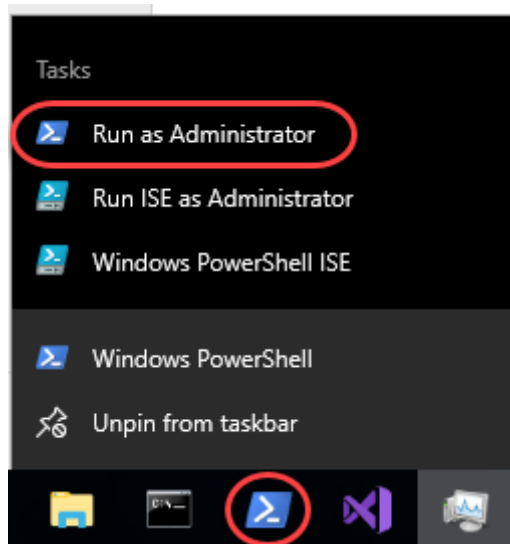
Registration script (PowerShell)

```
$ErrorActionPreference="Stop";If(-NOT ([Security.Principal.WindowsPrincipal]
[Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole
( [Security.Principal.WindowsBuiltInRole] "Administrator")){ throw "Run
command in an administrator PowerShell prompt";If($PSVersionTable.PSVersion
-lt (New-Object System.Version("3.0"))){ throw "The minimum version of
Windows PowerShell that is required by the script (3.0) does not match the
currently running version of Windows PowerShell." };If(-NOT (Test-Path
$env:SystemDrive\azagent')){mkdir $env:SystemDrive\azagent'; cd
$env:SystemDrive\azagent'; for($i=1; $i -lt 100; $i++)
{$destFolder="A"+$i.ToString();if(-NOT (Test-Path ($destFolder))){$mkdir
$destFolder;cd $destFolder;break;}}; $agentZip="$PWD\agent.zip";$DefaultProxy=
[System.Net.WebRequest]::DefaultWebProxy;$securityProtocol=@
();$securityProtocol+=
[Net.ServicePointManager]::SecurityProtocol;$securityProtocol+=
[Net.SecurityProtocolType]::Tls12;
[Net.ServicePointManager]::SecurityProtocol=$securityProtocol;$WebClient=New-
Object Net.WebClient; $Uri='https://go.microsoft.com/fwlink/?
linkid=2066756';if($DefaultProxy -and (-not $DefaultProxy.IsBypassed($Uri))
{$WebClient.Proxy= New-Object Net.WebProxy($DefaultProxy.GetProxy
($Uri).OriginalString, $True);}; $WebClient.DownloadFile($Uri, $agentZip);Add-
Type -AssemblyName System.IO.Compression.FileSystem;
[System.IO.Compression.ZipFile]::ExtractToDirectory( $agentZip,
"$PWD");.\config.cmd --deploymentgroup --deploymentgroupname "Local IIS"
--agent $env:COMPUTERNAME --runasservice --work '_work' --url
'http://vsalm:8080/tfs/' --collectionname 'PartsUnlimitedCollection'
--projectname 'Parts Unlimited' --auth Integrated; Remove-Item $agentZip;
```

Copy script to the clipboard

Run from an administrator PowerShell command prompt

11. From the taskbar, right-click **PowerShell** and select **Run as Administrator**.



12. Execute the script from the clipboard. Don't forget to click **Enter** after pasting it.


```

Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> $ErrorActionPreference="Stop";IF(-NOT ([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole] "Administrator")){ throw "Run command in an administrator PowerShell prompt";};If($PSVersionTable.PSVersion -lt (New-Object System.Version(3.0))){ throw "The minimum version of Windows PowerShell that is required by the script (3.0) does not match the currently running version of Windows PowerShell." };If(-NOT (Test-Path $env:SystemDrive\azagent)){mkdir $env:SystemDrive\azagent}; cd $env:SystemDrive\azagent; for($i=1; $i -lt 100; $i++){ $destFolder="A" + $i.ToString(); if(-NOT (Test-Path $destFolder)){mkdir $destFolder; cd $destFolder; break;}}; $agentZip="$PWD\agent.zip"; $defaultProxy=[System.Net.WebRequest]::DefaultWebProxy; $securityProtocol=[Net.ServicePointManager]::SecurityProtocol; $securityProtocol += [Net.SecurityProtocolType]::Tls12; [Net.ServicePointManager]::SecurityProtocol=$securityProtocol; $webClient=New-Object Net.WebClient; $url="https://go.microsoft.com/fwlink/?linkid=2066756"; if($defaultProxy -and (-not $defaultProxy.IsBypassed($url))){ $webClient.Proxy= New-Object Net.WebProxy($defaultProxy.GetProxy($url).OriginalString, $true);}; $webClient.DownloadFile($url, $agentZip); Add-Type -AssemblyName System.IO.Compression.FileSystem; [System.IO.Compression.ZipFile]::ExtractToDirectory($agentZip, "$PWD"); .\config.cmd --deploymentgroup --deploymentgroupname "Local IIS" --agent $env:COMPUTERNAME --runasser vice --work _work --url http://vsalm:8080/tfs/ --collectionname 'PartsUnlimitedCollection' --projectname 'Parts Unlimited' --auth Integrated; Remove-Item $agentZip;

Directory: C:\

Mode                LastWriteTime         Length Name
----                -
d-----          3/21/2019 12:42 PM             azagent

Directory: C:\azagent

Mode                LastWriteTime         Length Name
----                -
d-----          3/21/2019 12:42 PM              A1

```

- Accept the default options along the way. This will configure the agent to run as a service under the System account.

```

>> Connect:

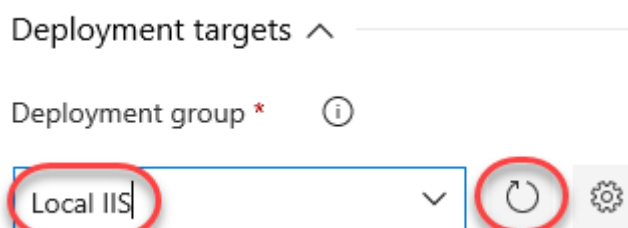
Connecting to server ...

>> Register Agent:

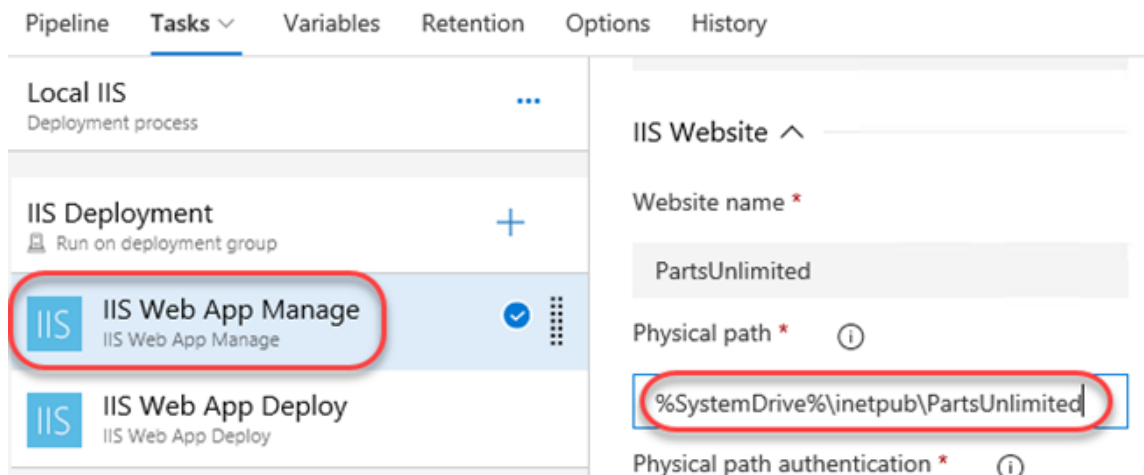
Scanning for tool capabilities.
Connecting to the server.
Enter deployment group tags for agent? (Y/N) (press enter for N) >
Successfully added the agent
Testing agent connection.
2019-03-21 19:43:16Z: Settings Saved.
Enter User account to use for the service (press enter for NT AUTHORITY\SYSTEM) >
Granting file permissions to 'NT AUTHORITY\SYSTEM'.
Service vstsagent.vsal.m.VSALM successfully installed
Service vstsagent.vsal.m.VSALM successfully set recovery option
Service vstsagent.vsal.m.VSALM successfully set to delayed auto start
Service vstsagent.vsal.m.VSALM successfully configured
Service vstsagent.vsal.m.VSALM started successfully

```

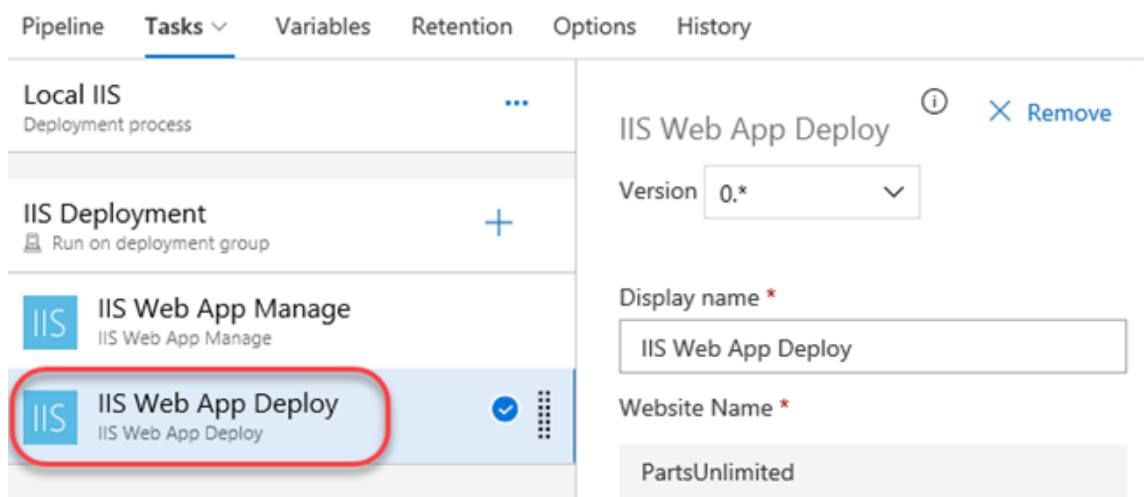
- Close the deployment groups browser tab.
- Return to the release pipeline and **Refresh** the deployment groups. Select **Local IIS**.



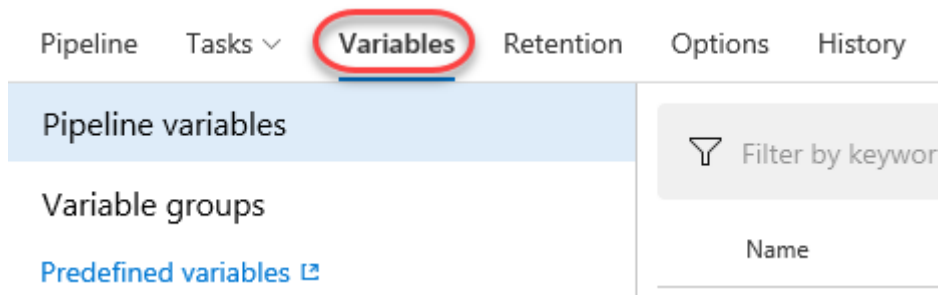
- Select **IIS Web App Manage**. This is the first task in the job that defines properties such as the local path to deploy to. Update it to **"%SystemDrive%\inetpub\PartsUnlimited"**.



17. Select the **IIS Web App Deploy** tab. This task actually deploys the site.



18. Select the **Variables** tab. This enables you to define pipeline-wide variables that can be managed centrally.



19. The **Retention** tab defines the policies for retaining releases.

Pipeline Tasks Variables **Retention** Options History

Local IIS
 Keep for 30 days, 3 good releases and keep artifacts.

Settings for Local IIS

 Days to retain a release * ⓘ

 Minimum releases to keep * ⓘ

☒ Retain associated artifacts ⓘ

20. The **Options** tab allows you to specify things like how releases are named.

Pipeline Tasks Variables Retention **Options** History

General

 Integrations

Description ⓘ

 Release name format ⓘ

21. The **History** tab lists release history.

Pipeline Tasks Variables Retention Options **History**

Changed By	Change Type	Changed Date	Comment
------------	-------------	--------------	---------

22. Click **Save** and confirm.

Save + Release ▾ ≡ View releases ...

Task 3: Invoking a manual release

1. From the **Release** dropdown, select **Create a release**.

Save + **Release** ▾ ...

+ Create a release
 + Create a draft release

2. You have the option to override the behavior of the pipeline, which we won't do now. Click **Create** to begin the release using the latest build.

Create a new release

Parts Unlimited-ASP.NET-CI - CD

Pipeline ^

Click on a stage to change its trigger from automated to manual.

 Local IIS

Stages for a trigger change from automated to manual. ⓘ

Artifacts ^

Select the version for the artifact sources for this release

Source alias	Version
Parts Unlimited-ASP.NET-CI	20190321.1

Create Cancel

3. Click the new release to view it.

[All pipelines](#) >  Parts Unlimited-ASP.NET-CI - CD

 Release **Release-1** has been created

4. You can visualize the release through the pipeline using the same kind of view. Click **In progress**.

Release

Manually triggered

by Sachin Raj
3/25/2019 1:30 PM

Artifacts

Parts Unlimited-ASP.NE...
[20190321.1](#)
 master

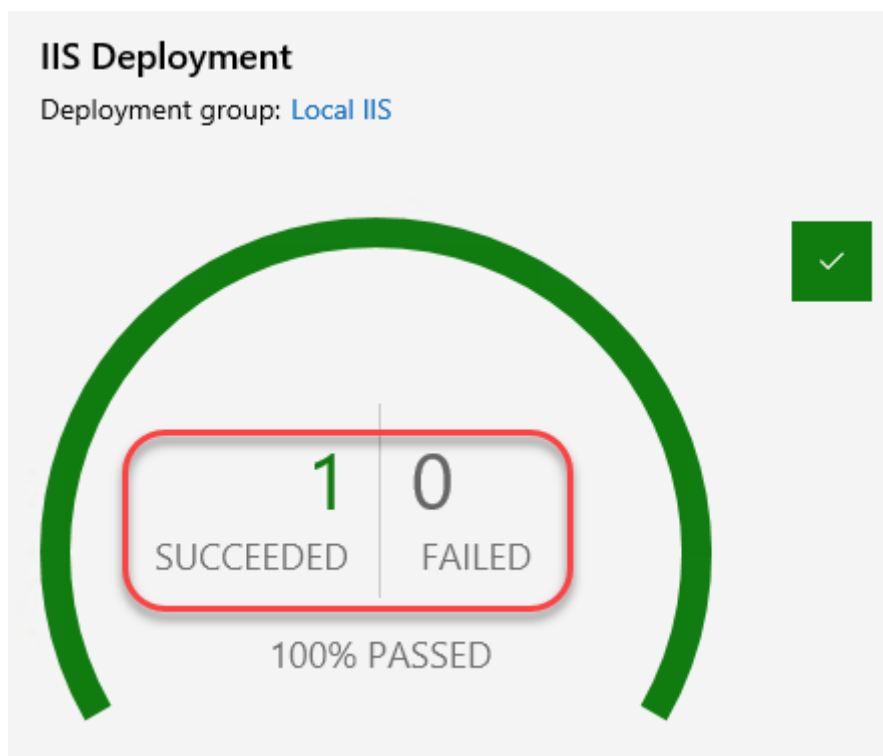
Stages

Local IIS

In progress

1 ☐
Target

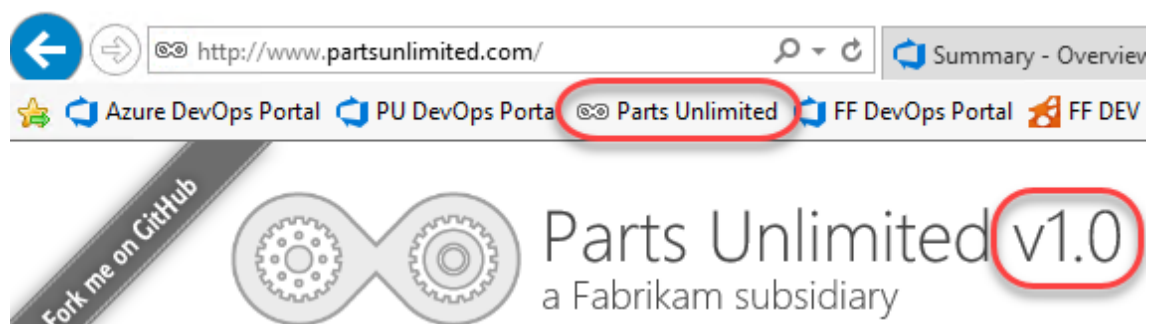
5. The release should complete pretty quickly since it's just a local deployment to IIS. Click the progress chart whether it's done or not.



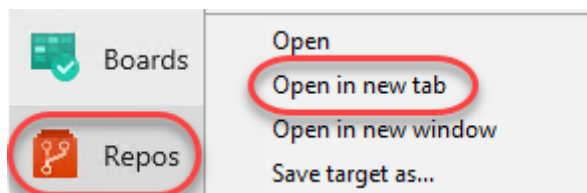
6. You can walk through each of the tasks to see the logs for what occurred.

VSALM		Started: 3/21/2019 1:06:05 PM
Target: VSALM		... 23s
✓	Initialize job · succeeded	3s
✓	Download Artifacts · succeeded	2s
✓	IIS Web App Manage · succeeded	5s
✓	IIS Web App Deploy · succeeded	11s

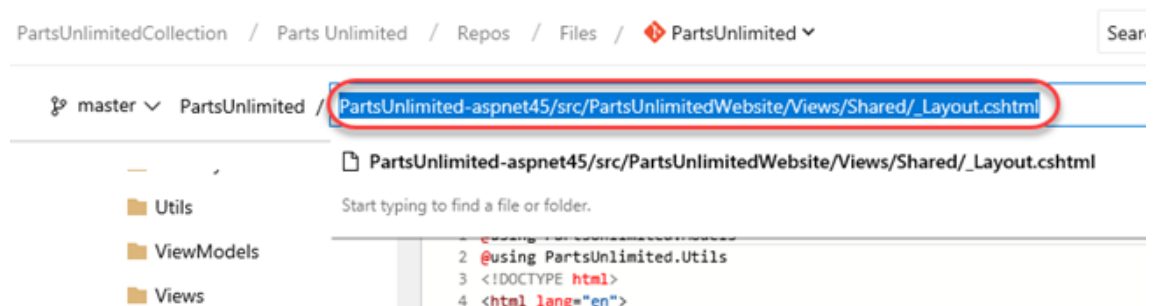
- Open a new browser tab and click the **Parts Unlimited** site shortcut. If you made a visible change to the site, it should be apparent here.



- Return to the release browser tab. Right-click **Repos** and select **Open in new tab**.



- Navigate to **PartsUnlimited-aspnet45/src/PartsUnlimitedWebsite/Views/Shared/_Layout.cshtml**.



- Click **Edit**.

Contents History Compare Blame **Edit** Rename Delete Download

- Make a cosmetic change by appending **"v2.0"** to the **h1** tag. Click **Commit** and confirm.

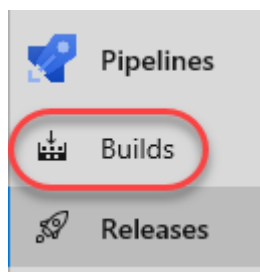
Contents Highlight changes **Commit...** Discard

```






36 <a class="block logo-image" href="@Url.Action("
37     Parts Unlimited v2.0</h1>
41     <h2>a Fabrikam subsidiary</h2>
42 </a>

```

- Return to the release browser tab and click **Builds**.



- Click the newly launched build to follow it.

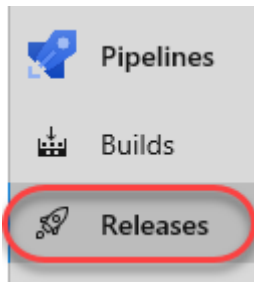
History Analytics*		Build #
Commit		
	Updated _Layout.cshtml CI build for Sachin Raj	 20190321.2
	Updated _Layout.cshtml CI build for Sachin Raj	  20190321.1

- Wait for the build to complete.



 **#20190321.2: Updated _Layout.cshtml**
 Triggered today at 1:22 pm for Sachin Raj  PartsUnlimited  master  8dbfd4d

Logs Summary Tests Database diff report Database script

- Navigate to the **Releases** view.






16. There should be a new release invoked by the completed build. Click it to open.

Releases		Analytics*	Edit		Create a release	...	
Releases			Created		Stages		
	Release-2 201903... ma...		2019-03-21 13:28		<input type="radio"/> Local IIS		
	Release-1 201903... ma...		2019-03-21 13:05		<input checked="" type="radio"/> Local IIS		


17. Like before, the release should quickly make its way through the pipeline and deploy to the local IIS.

Release

Continuous deployment
for  Sachin Raj
3/21/2019 1:28 PM

Artifacts

Parts Unlimited-A... 
20190321.2
master

Stages

Local IIS
 **Succeeded**
on 3/21/2019 1:32 PM

18. Refresh the tab open to the Parts Unlimited site and note that the v2.0 is now visible.

